Issue Handling Performance in Proprietary Software Projects

Aigerim Issabayeva*, Ariadi Nugroho† and Joost Visser†‡
*Tilburg University, The Netherlands – aigerim.issabay@gmail.com
†Software Improvement Group, The Netherlands – {a.nugroho, j.visser}@sig.eu
‡Radboud University Nijmegen, The Netherlands

***Abstract*—Software maintenance tasks are mainly related to fixing defects and implementing new features. Higher efficiency in performing such tasks is therefore going to reduce the costs of maintenance. A previous study involving open source systems has shown that higher software maintainability corresponds to faster speed in fixing defects [1]. In this paper we replicate the previous study by mining bug report data of three proprietary software projects. In one of the projects, a correlation between higher software maintainability and faster defect resolution is confirmed. The quality of issue handling process (e.g., issue registration accuracy and completeness, scope and complexity of issue workflow) should be considered in further research as it might explain the circumstances under which the correlation can be observed.**

***Keywords*-bugs; defects; resolution time; maintainability; productivity**

I. Introduction

The main activities in software maintenance are resolving software defects and implementing new functionality. Because these activities involve making changes to existing code, the maintainability of the code is believed to have a significant influence on the productivity in maintenance. In particular, it is logical to think that with higher maintainability, the issues found in a system will be resolved faster. In support of such assumption, previous research has shown that a correlation between software maintainability and issue handling performance exists [1]. The research was conducted on a data set comprised of open source projects. This paper replicates the previous work using data from proprietary software projects and reports preliminary results.

To measure issue handling performance, the same approach defined in [1] is used. This approach rates the issue handling performance using a star rating with a scale from 0.5 to 5.5. Software maintainability is measured using a maintainability model developed by the Software Improvement Group (SIG) [2], which is based on the ISO/IEC 9126 international standard on software quality, including software maintainability [3]. This quality model has been used by SIG to perform software risk assessments, software monitoring, and—in collaboration with TÜViT [4]—certifications of software products' maintainability.

II. Issue Handling Performance

In this section we further discuss our approach in measuring and assigning rating to issue handling performance.

A. Issue Resolution Time

The issue resolution time is the time taken to close an opened issue, estimated by the time period between issue being opened and it being closed. It is important to differentiate between an issue being resolved and an issue being closed, as an issue can be closed without being resolved (e.g., duplicate and non-reproducible issues). We only take into account issues that were closed after the problems were resolved. The following are the typical stages of an issue lifecycle: New - Assigned - Resolved - Closed.

Issue tracking systems generally have customizable issue states flow. Hence, it is difficult to correctly establish the time it has really taken for a developer to solve an issue. Therefore, we take the time between the opening of the issue and its closing. Also, if the issue was reopened, the issue resolution time is the time the issue was in an open state, meaning that we exclude the time that the issues was considered to be closed.

Previous work used different ways of measuring the time to fix defects. Kim and Whitehead used the time between bug-introducing changes and their corresponding fixes to measure the time to fix bugs [5]. Weiss et al. used the information about the effort spent (in person hours) to fix bugs that is recorded in the bug reports [6]. The bug database that we received does not record files that were changed to fix bugs nor the effort spent to fix them. Therefore, we could not compare the results of our method to those of the aforementioned work.

B. Issue Resolution Rating

Calculating issue resolution time is useful for understanding the distribution of the time required to resolve issues in a project. However, issue resolution time alone does not help much to understand whether the issue handling is efficient, particularly in comparison to other projects. To have a more meaningful measure of issue handling performance, Luijten et al. [1] proposed a way to rate issue resolution time based on issue risk profiles.

A risk profile is an assignment of issue resolution time (measured in days) into four risk categories, namely low, moderate, high, and very high risk, based on certain thresholds (see Table I). For example, if an issue has been resolved in less than four weeks, it would be assigned to the low category, whereas an issue that was resolved in more than

Table I
THRESHOLDS FOR RISK CATEGORIES OF ISSUE RESOLUTION TIME

Category	Thresholds	
Low	0 - 28 days	(4 weeks)
Moderate	28 - 70 days	(10 weeks)
High	70 - 182 days	(6 months)
Very high	182 days or more	

Table II
THRESHOLDS FOR QUALITY RATINGS OF ISSUE RESOLUTION TIME

Rating	Moderate	High	Very High
*****	8.3%	1.0%	0.0%
****	14%	11%	2.2%
***	35%	19%	12%
**	77%	23%	34%

10 weeks but less than 6 months would be of a high risk category. Every issue is assigned to the risk category in the same manner. In a project where the risk profile is {60, 20, 15, 5}, 60% percent of the issues fall into the low risk categories, and 20%, 15%, 5% fall into the moderate, high, and very high risk categories respectively.

Once all the issues are assigned to a risk category, the issue resolution rating of a project is calculated by considering the percentage of issues belonging to each risk category as in Table II. For example, a system with less than 8.3% of the issues belonging to the moderate risk category, less than 1% belonging to the high risk category, and none belonging to the very high risk category, will achieve a 5-star rating. These rating thresholds are based on a benchmark of around 100 releases of various open source projects [7]. The rating thresholds are calibrated so that 5% of the system releases score five stars, 30% four stars, 30% three stars, 30% two stars and the last 5% receive one star rating.

III. CASE STUDIES

The data analysis has been conducted on the data obtained from three proprietary projects. For confidentiality reasons, the projects are named Project A, Project B and Project C. Project A and Project B concern a banking system, while Project C deals with an operational system used to manage public transport. The three projects are generally maintenance projects, with the slight exception of Project C wherein a re-architecting was started before and still carried out during the execution of this study. Perfective maintenance was also performed in Project A and B, which was mainly targeted at improving code maintainability. The characteristics of each project is summarized in Table III.

Issues generally consist of defects and enhancements. However, in this paper we focus only on issue categorized as defect because the number of enhancements in the analyzed projects is relatively small. Therefore, hereafter we use the term issue to refer *only* to defect.

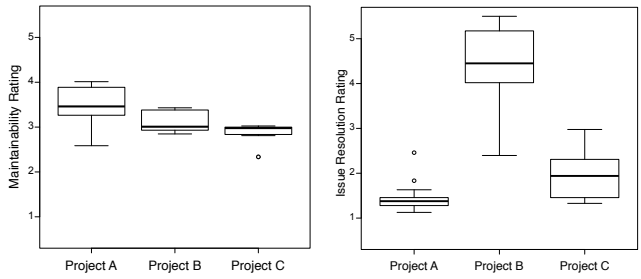

Figure 1. Maintainability Rating and Issue Resolution Rating across Projects

A. Descriptive Statistics

Issue resolution ratings and maintainability ratings are calculated per system snapshot. An issue is linked to a particular system snapshot based on the matching between the date the issue is resolved and the date of a system snapshot. More precisely, for each snapshot, its corresponding issues are those that are resolved between the date of that snapshot and the next.

Initially all available snapshots in the projects were considered. However, for the correlation analysis we need to maintain independence between snapshots. Therefore, we set a criteria that there is at least one-month distance between two consecutive snapshots. Furthermore, snapshots that do not correspond to any issues are excluded from the correlation analysis. In the end, the number of snapshots in Project A, B, and C are 14, 21, and 12 respectively.

Figure 1 shows two boxplots of maintainability rating and issue resolution rating across projects. As mentioned previously, maintainability ratings are determined based on the maintainability model defined in [2].

The most interesting point to note from the boxplots is the median values (indicated by horizontal lines printed in bold inside the boxes). We can see that the median values of the maintainability rating across projects are quite similar—roughly 3 star. Nevertheless, the medians of issue resolution rating across projects look somewhat different in that the median of Project B is far above Projects A and C. Essentially, this result shows that in Project B issues are being resolved much faster than in Project A and C.

The median values of the maintainability rating and issue resolution rating if the data sets of the three projects are combined are 3.0 star and 2.3 star respectively.

B. Correlation Analyses

The objective of the analysis is to find out whether there is a correlation between the maintainability ratings and the issue resolution ratings. The Spearman's rank correlation test is used to evaluate the correlation. We use statistical confidence of 95% ($p \leq 0.05$) to qualify significant correlations.

Table IV presents the results of correlation analyses between maintainability ratings and issue resolution ratings

Table III
PROJECTS' CHARACTERISTICS

Characteristic	Project A	Project B	Project C
Size (SLOC)	166,463	76,893	701,815
Data available since	2008	2008	2004
Main technology	Java	Java	Java
Submitted issues	3,807	2,045	7,287
Resolved issues (%)	832 (22%)	1,421 (69%)	5,079 (70%)
Resolved issues per KLOC	5	18	7

Table IV
RESULTS OF SPEARMAN'S CORRELATION ANALYSES BETWEEN ISSUE RESOLUTION RATING AND MAINTAINABILITY RATING

Metric	All Projects	Project A	Project B	Project C
Maintainability	-	-	0.579**	-

** *indicates* $p \leq 0.01$*; two-tailed.*

in the three projects. Performing correlation analysis on the combined data set of the three projects does not show a significant correlation between issue resolution rating and maintainability rating. However, performing the analyses on the data set of the individual projects reveals a significant correlation in Project B ($r = 0.579$; $p \leq 0.01$). A correlation of this magnitude suggests that maintainability rating explains 33% (r-square) of the variability in issue resolution rating. A positive correlation indicates that issue resolution rating increases as the maintainability rating increases.

To explain why a significant correlation exists only in Project B, we inspect the correlation between maintainability rating and issue resolution rating using a scatterplot as shown in Figure 2. The figure shows that the red circles (Project B) are quite separated from the rest of the data points—more specifically, most issues in Project B have resolution ratings higher than 3.5 star. Consistent with the result of the correlation test, the figure also shows a trend indicating a positive correlation between maintainability rating and issue resolution rating in Project B.

C. Assessments of Defect Handling Process

The fact that most issues in Project B have issue resolution ratings higher than most issues in the other two projects may indicate a substantial difference in the process through which issues are resolved. One way to assess issue handling process is by looking at the Issue Churn View [8].

Figure 3, 4, and 5 show the issue churn of Project A, B, and C respectively. The light green and light red (the symmetry) both represent the same issues that were opened and closed in the same month. Dark green represents issues closed in the month, but opened for longer than a month. Dark red represents old open issues. Finally, dark grey represents recent backlog (less than 6 months), and light grey mature backlog (more than 6 months).

Looking at Figure 3 we can determine that not many

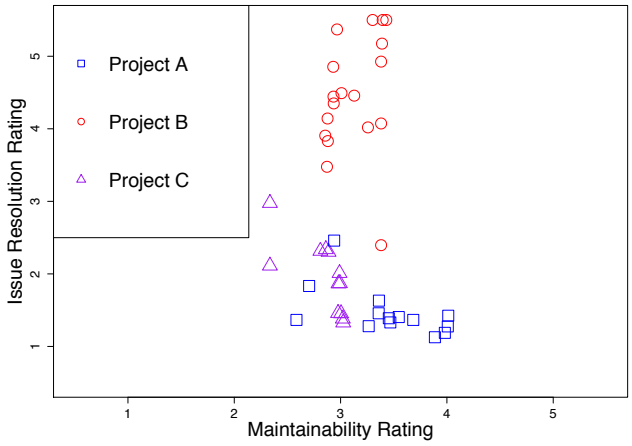

Figure 2. A scatterplot of Maintainability Rating and Defect Speed Rating

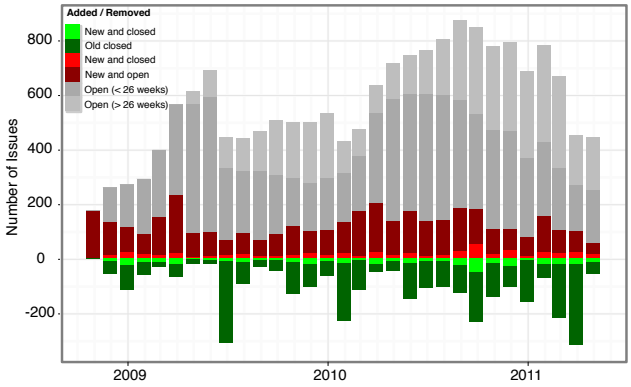

Figure 3. Issue Churn of Project A

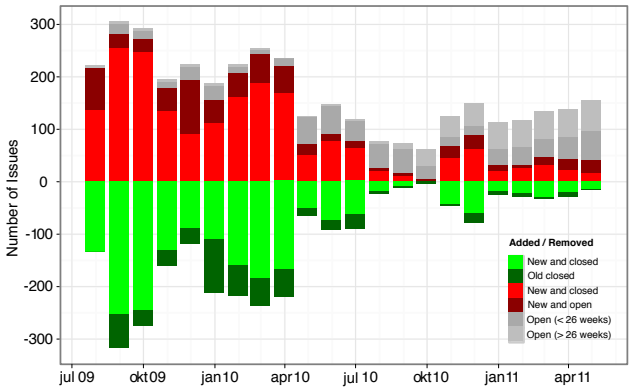

Figure 4. Issue Churn of Project B

issues are being solved throughout the lifecycle of Project A. The backlog is considerably greater than the amount of issues being closed, which actually corresponds to the fact that only 22% of the issues were resolved. In the issue churn of Project B shown in Figure 4 we can observe large symmetries of incoming and outgoing issues in the same

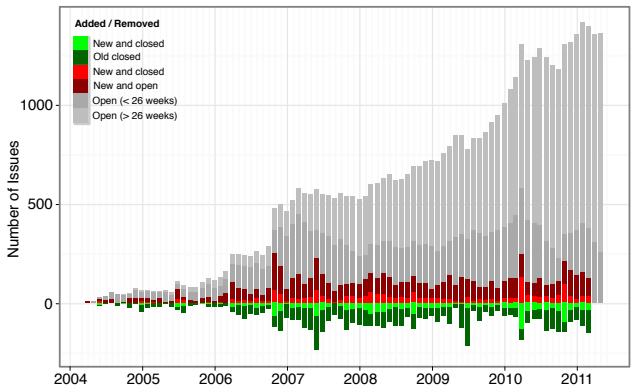

Figure 5. Issue Churn of Project C

month, which results in a considerably low backlog. The backlog stays low from the mid of 2009 to beginning of 2010, leveling out and slowly growing in 2011.

The issue churn of Project C as presented in Figure 5 shows a longer timespan (2004 - 2011). In the beginning of 2004 until early 2006 the project seemed to handle issues quite well. However, in November 2004 the backlog increased substantially and kept on growing thereafter. The project's ability to resolve issues did not seem to keep up with the amount of incoming issues—a similar pattern that is also observed in Project A. In particular, we see that in Project A and C many of the submitted issue were seldom resolved in the same month, which later has led to a growing backlog.

Issue handling behaviors observed from the issue churn views might reflect the underlying issue handling processes in the three projects. In particular, the issue handling processes of Project A and C do not seem to allow efficient issue handling as in Project B. This difference in efficiency can be explained by some factors such as the number of available resources and the quality of the issue handling process (e.g., in terms of complexity and length of the process). Every project might have different requirements for the issue handling process (e.g., stricter validation phase). Accounting for the effect of issue handling process might give better insights about the interrelationship amongst software maintainability, issue handling performance, and issue handling process.

IV. Conclusion

In this paper the correlation between issue handling performance and software maintainability is assessed using empirical data from three proprietary software projects. Correlation analyses are performed on the data set of the three projects combined as well as on the individual project data sets. The findings are summarized as follows:

- Analyses on the combined data set of the three projects do not show any significant correlation between issue handling performance and software maintainability.
- Performing analyses on individual data sets reveals a positive and statistically significant correlation in one of the projects.
- A significant correlation is found in a project that has a high issue handling performance (rated 4.5 star).

For future work we plan to define a way to quantify the maturity of issue handling process and assess whether process maturity has a confounding effect on the correlation between software maintainability and issue handling performance. Additionally, since open source projects typically rely more heavily than proprietary projects on the issue tracking system for communication among developers, the relation observed in the previous study on open source projects [1] may have been enabled by the relatively high issue handling maturity of these projects. Therefore, a future study should include both open source and proprietary projects and compare their maturity levels.

References

[1] B. Luijten, J. Visser, and A. Zaidman, "Faster defect resolution with higher technical quality of software," in *Proceedings of the 4th international workshop on software quality and maintainability*, 2010.

[2] I. Heitlager, T. Kuipers, and J. Visser, "A practical model for measuring maintainability," in *Quality of Information and Communications Technology, 2007. QUATIC 2007. 6th International Conference on the*. IEEE, 2007, pp. 30–39.

[3] International Organization for Standardization, "ISO/IEC 9126-1: Software engineering - product quality - part 1: Quality model," 2001.

[4] R. Baggen, K. Schill, and J. Visser, "Standardized code quality benchmarking for improving software maintainability," in *Proceedings of the 4th International Workshop on Software Quality and Maintainability*, 2010.

[5] S. Kim and E. Whitehead Jr, "How long did it take to fix bugs?" in *Proceedings of the 2006 international workshop on Mining software repositories*. ACM, 2006, pp. 173–174.

[6] C. Weiss, R. Premraj, T. Zimmermann, and A. Zeller, "How long will it take to fix this bug?" in *Proceedings of the Fourth International Workshop on Mining Software Repositories*. IEEE Computer Society, 2007, p. 1.

[7] D. Bijlsma, M. Ferreira, B. Luijten, and J. Visser, "Faster issue resolution with higher technical quality of software," *Software Quality Journal*, pp. 1–21, 2011.

[8] B. Luijten, J. Visser, and A. Zaidman, "Assessment of issue handling efficiency," in *Proceedings of the 7th IEEE Working Conference on Mining Software Repositories*. IEEE, 2010, pp. 94–97.